Identity-Based Key Agreement for Blockchain-Powered Intelligent Edge

Jie Zhang and Futai Zhang

***Abstract*—In the new paradigm of blockchain-powered intelligent edge, the key agreement is a significant problem which has not been extensively studied so far. Existing key agreement protocols in the traditional public-key setting are usually too complicated and heavy for edge and end devices. Besides, most protocols in use do not have effective measures to resist side-channel attacks which are increasingly threatening cloud servers, edge devices and end devices. Identity (ID)-based protocols can be conveniently implemented in the blockchain-powered intelligent edge. Several leakage-resilient ID-based protocols which can resist side-channel attacks have been proposed. However, they all involve time-consuming pairing computations. Besides, none of them address side-channel attacks to the key generation center (KGC).**

This paper designs and realizes two novel ID-based key agreement protocols for the blockchain-powered intelligent edge, including an extended Canetti-Krawczyk (eCK) secure ID-based authenticated key agreement (AKA) protocol and a continuous after-the-fact leakage-resilient eCK (CAFL-eCK) secure ID-based AKA protocol. Both protocols do not involve any heavy pairing computation. Besides, the second one can resist side-channel attacks to the KGC and the communicating parties. A hybrid implementation of the two protocols can achieve high efficiency and strong security at the same time in blockchain-powered intelligent edge environments. This is demonstrated via a use case of a blockchain-powered smart home.

***Index Terms*—identity-based key agreement, side-channel attacks, leakage resilience, blockchain, intelligent edge, intelligent Internet of Things.**

I. Introduction

RECENT research on Intelligent Internet of Things (IIoT) emphasizes the significance of the intelligent edge layer. As the middle layer between the cloud layer and the end layer, intelligent edge can undertake computing tasks for smart end devices on behalf of the cloud center. Compared with traditional IoT paradigm which uploads data from end devices to the remote cloud center for processing, the new paradigm has a number of advantages, such as reducing response time, alleviating security and privacy issues during uploading, and so on. However, it also brings the problem for the cloud center to trace data. Under this background, blockchain is introduced into the intelligent edge layer to enable traceability. The integration of blockchain and intelligent edge are studied in some recent works [1], [2], [3], [4], [5], [6] which have established the framework of blockchain-powered intelligent edge.

Key agreement is a significant problem in the blockchain-powered intelligent edge. It establishes secret keys that underlie the security of communications crossing and within layers. However, there are several challenges. First, the most frequently used key agreement methods [7] are based on traditional public key cryptosystem which explicitly authenticates a user's public key by a certificate issued by a fully trusted certificate authority (CA for short) [8]. Second, most key agreement protocols in use can not resist side-channel attacks which are increasingly threatening cloud servers, edge devices and end devices [9], [10], [11], [12].

The identity-based (ID-based) cryptosystems [13] are natively suitable for the blockchain-powered intelligent edge environments. They do not require any digital certificate and thereby are much more convenient than traditional public-key cryptosystems. The public keys are derived from users' identity information, and the private keys are computed by a key generation center (KGC). This can be easily realized in blockchain-powered intelligent edge environments by setting the cloud server as KGC and using the identities of intelligent edge and end devices to derive their public keys.

For application in the blockchain powered intelligent edge environments, a key agreement protocol should be lightweight and leakage resilient. Although some existing pairing-free and elliptic curve cryptography (ECC)-based protocols are lightweight for smart edge and end devices [14], [15], [16], [17], most of them cannot resist the side-channel attacks. We noticed that several key agreement protocols secure under side-channel attacks [11], [12] were proposed in recent years. But they are all pairing-based requiring heavy computations, and none of them consider the side-channel attacks to KGC. So far, the most suitable one which can overcome the aforementioned challenges in the blockchain powered intelligent edge environments is still unavailable in the literature.

Therefore, this paper focuses on the design and implementation of ID-based key agreement protocols for the blockchain-powered intelligent edge. Both security and performance in the senaro of blockchain-powered intelligent edge are fully taken into account. The main contributions are summarized as follows. Firstly, two ID-based authenticated key agreement (AKA) protocols are put forward. Protocol I is secure under

Manuscript received August 2, 2021; revised September 7, 2021. (Corresponding author: Futai Zhang)

This work was partially supported by the National Natural Science Foundation of China under Grant No. 62172096 and No. 62002296; the Natural Science Foundation of Jiangsu Province under Grant No. BK20200250; the XJTLU Key Programme Special Fund under Grant No. KSF-E-54.

J. Zhang is with the School of Advanced Technology, Xi'an Jiaotong-Liverpool University, Suzhou, China. E-mail: jie.zhang01@xjtlu.edu.cn

F. Zhang is with Fujian Provincial Key Laboratory of Network Security and Cryptology, College of Computer and Cyber Security, Fujian Normal University, Fuzhou 350117, China. E-mail: ffttzhang@sina.com; futai@fjnu.edu.cn

the extended Canetti-Krawczyk (eCK) model. Protocol II is the leakage-resilient version of Protocol I. It has very strong provable security under the continuous after-the-fact leakage-resilient eCK (CAFL-eCK) model. It can resist side-channel attacks to both the KGC (i.e., cloud centers) and users (i.e., edge devices and end devices). Secondly, prototypes for the two proposed protocols are realized, and a set of experiments are carried out to study their performance. The experimental results show that both protocols are acceptable for commonly used devices in blockchain-powered intelligent edge environments. Finally, a hybrid implementation of the two protocols are demonstrated via a use case of blockchain-powered smart home. It shows that the two protocols can achieve high efficiency and strong security at the same time in blockchain-powered intelligent edge environments.

The rest of this paper is organized as follows. Section II reviews some representative ID-based AKA protocols. Section III presents related mathematical and cryptographical background. Section IV presents the system model. Section V, VI, VII propose two ID-based AKA protocols, prove their security and study their performance respectively. Section VIII demonstrates the application of the two protocols through a use case. Finally, Section IX concludes this paper.

II. Related Work

This section reviews existing ID-based AKA and leakage-resilient AKA. Section II-A chronologically reviews ID-based AKA protocols from early-stage ones to some recent ones. It shows that the proposed ID-based AKA protocols have innovation and are valuable development to the research field. Section II-B reviews leakage-resilient models for AKA from weaker to stronger ones. It shows that the prototypes and use case of the proposed protocols narrow the gap between theory and practice in the research field.

A. ID-Based AKA

Since Sharmir proposed the concept of ID-based cryptography in 1984 [13], a large number of ID-based AKA protocols were presented. This section gives a review of some representative two-party ID-based AKA protocols in the literature.

1) Early-Stage ID-Based AKA: The early-stage ID-based AKA protocols were proposed in 1980s. The first ID-based AKA was published by Okamoto in 1986 [18]. Its KGC setup and private key extract are based on the RSA problem, and key agreement is based on the discrete logarithm problem. Another famous protocol was proposed by Gunther in 1989 [19]. It uses the ElGamal signature in private key extract. These early-stage protocols are pioneers of ID-based AKA though they have some drawbacks in terms of design and security [20].

2) ID-Based AKA from Bilinear Pairings: After Boneh and Franklin presented the first practical ID-based encryption scheme from Weil pairing in 2001 [21], a number of ID-based AKA protocols from bilinear pairing were proposed. In 2002, Smart published the first ID-based AKA protocol based on Weil pairing [22]. In 2003, Chen and Kudla improved Smart's protocol in terms of efficiency and security [23]. In 2005, McCullagh and Barreto designed a new efficient ID-based AKA which can be instantiated in either escrowed or escrowless mode [24]. These pairing-based protocols have a similar construction in KGC setup and private key extract. The main difference lies in the computation of the shared secrets. The problem of these protocols is involving time-consuming pairing computations.

3) ID-Based AKA without Bilinear Pairings: Paring-free ID-based AKA protocols remove the computations of bilinear pairing and thus improve the efficiency. Some typical protocols were proposed by Fiore and Gennaro in [25], Cao et al. in [14], Xie and Wang in [26], Sun et al. in [15], Ni et al. in [27], Bala et al. in [16], and Islam and Biswas in [28]. These protocols have a very similar design in KGC setup and private key extract, and all use Diffie-Hellman key exchange or its elliptic curve version in key agreement. The main difference lies in session key computation, which leads to different provable security in different security models.

4) ID-Based AKA in Recent Years: In recent years, several new ID-based AKA protocols were proposed to achieve better applicability for different application scenarios of IoT. Some lightweight protocols are designed to reduce computing burdens on limited IoT devices. For example, Zhang et al. designed an ID-based AKA protocol for disaster scenarios in [17]. Their protocol is pairing-free, has a low computing requirement on capability-limited devices, and thereby is friendly to limited IoT devices. Kumar and Saxena also proposed a pairing-free ID-based AKA for resource-constrained devices [29]. Some protocols with anonymity were proposed for scenarios such as smart grid, mobile edge computing and fog computing where privacy is required. For example, Lei et al. proposed an ID-based AKA protocol with anonymity in the heterogeneous wireless networks [8]. Mahmood et al. proposed an ID-based AKA with both anonymity and untraceability for smart grid advanced metering infrastructure [30]. Patonico et al. proposed an ID-based and anonymous AKA protocol for fog computing [31]. Hassan et al. proposed an ID-based AKA protocol for multi-server environment with anonymity [32]. Jia et al. designed an ID-based anonymous AKA protocol for mobile edge computing environment [33]. Leakage attacks are not considered in these protocols.

5) Leakage-Resilient ID-Based AKA: The security of all the aforementioned protocols is based on some mathematically hard problems and the assumption that the long-term private keys are secure from attacks. However, the emerging side-channel attacks can leak information of the long-term secrets in physical manners. Under this situation, leakage-resilient cryptography is established as a young branch that studies cryptographic primitives resisting side-channel attacks. In recent year, several leakage-resilient ID-based AKA protocols were proposed. Ruan et al. proposed a pairing-based ID-based AKA protocol that is secure in bounded-leakage and extended-Canetti–Krawczyk (eCK) security model [11]. Wu et al. designed an ID-based AKA with leakage-resilience in the continuous-leakage eCK model [12]. Although these works can resist some types of side-channel attacks, they are not suitable for the blockchain-powered intelligent edge for two reasons. Firstly, all of these protocols involve heavy pairing

computations. Secondly, they do not consider side-channel attacks to the KGC.

6) Summary: Although ID-based AKA is not quite new research in cryptography, it is still active in both academia and industry. Many new protocols are designed in recent years to achieve better applicability in innovative application scenarios and to achieve higher security against emerging attacks. However, the most suitable one for the blockchain-powered intelligent edge is still unavailable.

This paper presents two pairing-free ID-based AKA protocols that are more applicable than existing ones in blockchain-powered intelligent edge environments. Table I compares the proposed protocols with some existing ID-based AKA protocols in terms of efficiency (involving pairing or not), leakage-resiliency, and provable security. It shows that the proposed Protocol I has the strongest provable security among all non-leakage-resilient protocols, and Protocol II has the strongest provable security among all leakage-resilient protocols[1].

B. Leakage-Resilient AKA

Leakage-resilient model for AKA and representative works are introduced as follows.

1) Bounded Leakage: The bounded leakage model assumes the amount of leakage is limited to an upper bound in the entire execution period. D. Moriyama and T. Okamoto [43] proposed a PKI-based AKA protocol which is λ-leakage resilient eCK (BL-eCK) secure, where λ is the upper bound for the leakage of long-term private keys.

2) Continuous Leakage: The continuous leakage model assumes the leakage is limited to an upper bound in each execution, but there is no bound on the overall leakage. It is stronger than the bounded leakage. J. D. Wu et al. [12] proposed an ID-based AKA protocol which is continuous leakage-resilient eCK (CL-eCK) secure.

3) After-the-fact Leakage: After-the-fact leakage refers to leakage which happens after the challenge is given to the adversary. There are bounded after-the-fact leakage (BAFL) and continuous after-the-fact leakage (CAFL) models which are more stronger than the bounded and continuous leakage models. J. Alawatugoda et al. studied AKA in both BAFL and CAFL models and proposed BAFL-eCK secure [38] and CAFL-eCK secure [37] AKA protocols.

4) Summary: The strongest leakage-resilient model for AKA is CAFL-eCK model. Although there are some CAFL-eCK secure AKA protocols, a CAFL-eCK secure ID-based AKA protocol which is the most suitable for our scenario is still unavailable in the literature. Besides, most work on leakage-resilient AKA focus on the design and security proof of protocols, but prototypes or application guidelines are still unavailable. Therefore, our work which involves the design, security proof, prototype and use case is a meaningful development to the feild of leakage-resilient AKA and can narrow the gap between theory and practice.

[1]eCK is the strongest security model for non-leakage-resilient AKA protocols. CAFL-eCK is the strongest security model for leakage-resilient AKA protocols.

III. Preliminaries

This section presents related mathematical and cryptographical backgrounds for our new protocols. Symbols used in this paper are explained in Table II.

A. Elliptic Curve Group and Hard Problems

Let F_p be a finite field of integers modulo a prime number p, and E be an elliptic curve defined by the following equation

$$y^2 = x^3 + ax + b$$

where $a, b \in F_p$ and $4a^3 + 27b^2 \neq 0 \mod p$.

An elliptic curve group G consists of all points on E together with the point at infinity. The following two operations are defined in G:

- Point addition of two points P_1 and P_2 is denoted by $P_1 + P_2$ which returns a third point in G.
- Scalar multiplication between an integer t and a point P is denoted by $[t]P$ which returns a third point in G. It can be computed as

$$[t]P = \underbrace{P + P + \cdots + P}_{t}$$

For vectors $\boldsymbol{v} = (v_1, \cdots v_m)$ and $\boldsymbol{w} = (w_1, \cdots, w_m)$, we define

$$[\boldsymbol{v}]P = ([v_1]P, \cdots, [v_m]P)$$

and

$$[\boldsymbol{v}]([\boldsymbol{w}]P) = [\boldsymbol{v} \cdot \boldsymbol{w}]P = [v_1 w_1 + \cdots + v_m w_m]P.$$

Here $\boldsymbol{v} \cdot \boldsymbol{w} = v_1 w_1 + \cdots + v_m w_m$ denotes the inner product of vectors $\boldsymbol{v}$ and $\boldsymbol{w}$.

The following problems are assumed to be hard over G of order n with a generator P [34].

1) Elliptic Curve Computational Diffie–Hellman (ECCDH) Problem: For some random $a, b \in Z_n$, given $(P, G, [a]P, [b]P)$, output $[ab]P$.

2) Elliptic Curve Decisional Diffie–Hellman (ECDDH) Problem: For some random $a, b, c \in Z_n$, given $(P, G, [a]P, [b]P, [c]P)$, output 1 if $[c]P = [ab]P$ and 0 otherwise.

3) Elliptic Curve Gap Diffie–Hellman (ECGDH) Problem: For some random $a, b \in Z_n$, given $(P, G, [a]P, [b]P)$ and an oracle that solves the ECDDH problem on G, output $[ab]P$.

B. ID-Based AKA

An ID-based AKA is composed of three phases executed by KGC and users.

- Phase 1: KGC setup. KGC initializes the system parameters and the master public and private keys. The master public key and system parameters are published to all users. The master private key is securely kept by KGC.
- Phase 2: Extract. Firstly, a user submits the private key extract request with identity ID to KGC. Then, KGC generates the private key using ID, the master private key, and a random value. Finally, KGC returns the private key to the user through some secure channels. The corresponding public key can be derived from ID.

TABLE I: Comparison

Protocol	Involving pairing	Leakage-resiliency on users	Leakage-resiliency on KGC	Provable security
[18]	×	×	×	×
[19]	×	×	×	×
[22]	✓	×	×	×
[23]	✓	×	×	BR[39]
[24]	✓	×	×	BR
[14]	×	×	×	mBR[40]
[15]	×	×	×	eCK
[16]	×	×	×	Modified eCK[42]
[25]	×	×	×	CK[41]
[26]	×	×	×	CK
[27]	×	×	×	eCK
[28]	×	×	×	×
[17]	×	×	×	mBR
[29]	×	×	×	×
[8]	×	×	×	eCK
[30]	×	×	×	CK
[31]	×	×	×	CK
[32]	✓	×	×	Random oracle
[33]	✓	×	×	Random oracle
[11]	✓	✓	×	BAFL-eCK
[12]	✓	✓	×	CL-eCK
Protocol I	×	×	×	eCK
Protocol II	×	✓	✓	CAFL-eCK

TABLE II: Symbols

Symbol	Meaning
F_p	The finite field with the prime order p.
E	The elliptic curve.
G	The elliptic curve group of E.
$[t]P$	The scalar multiplication between an integer t and a point P on E.
$[\boldsymbol{v}]P$	The scalar multiplication between a vector $\boldsymbol{t} = (t_1, \cdots, t_m)$ and a point P on E. The output is the vector $([t_1]P, \cdots, [t_m]P)$
P	A generator of G.
Z_n	The set $\{0, \cdots, n-1\}$ with respect to addition modulo n where n is the order of G and its generator P.
$\|$	The concatenation of bit strings.
x	The master private key of KGC.
P_{pub}	The master public key of KGC.
H_1	A cryptographic secure hash function $\{0,1\}^* \times G \rightarrow Z_n$.
H_2	A cryptographic secure hash function $\{0,1\}^* \times \{0,1\}^* \times G \times G \times G \times G \rightarrow \{0,1\}^k$.

- Phase 3: AKA. Two users generate ephemeral private and public keys, exchange ephemeral public keys via public channels, and compute a shared session key based on their long-term and ephemeral private and public keys. The long-term private keys are generated in Phase 2.

C. Leakage-Resilient Method from Inner-Product

This method consists of an encoding protocol [35] which randomly encodes a secret $s \in Z_n$ into two states $\boldsymbol{s_L}, \boldsymbol{s_R} \in Z_n^{*m} \backslash \{(0^m)\}$ such that the inner product $\boldsymbol{s_L} \cdot \boldsymbol{s_R} = s$, and a refreshing protocol which securely refreshes the two states in the presence of leakage.

1) Leakage-Resilient Encoding: $Encode^m_{Z_n^*}(s)$ randomly selects $\boldsymbol{s_L} \in Z_n^{*m} \backslash \{(0^m)\}$, samples $\boldsymbol{s_R} \in Z_n^{*m} \backslash \{(0^m)\}$, and outputs $(\boldsymbol{s_L}, \boldsymbol{s_R})$

The leakage-resilience of the encoding protocol is stated in Theorem 1.

Theorem 1: Let $m > 1$ and $|Z_n| = \Omega(m)$. For any $1/2 > \delta > 0, \gamma > 0$, $Encode^m_{Z_n^*}(s)$ is (λ, ϵ)-secure where $\lambda = (1/2 - \delta)m \log |Z_n| - \log \gamma^{-1}$ and $\epsilon = 2(|Z_n|^{2/3 - m\delta}) + |Z_n|\gamma$.

2) Leakage-Resilient Refreshing: $Refresh^m_{Z_n^*}(\boldsymbol{s_L}, \boldsymbol{s_R})$ refreshes $(\boldsymbol{s_L}, \boldsymbol{s_R})$ into $(\boldsymbol{s_L}', \boldsymbol{s_R}')$ as follows:

1. Refresh $\boldsymbol{s_R}$:
   - Sample two vectors $\boldsymbol{a_R}, \boldsymbol{b_R} \in Z_n^{*m} \backslash \{(0^m)\}$ such that $\boldsymbol{a_R} \cdot \boldsymbol{b_R} = 0$
   - Generate a random non-singular matrix $\boldsymbol{M} \in Z_n^{*m \times m}$ such that $\boldsymbol{s_L} \cdot \boldsymbol{M} = \boldsymbol{a_R}$
   - Set $\boldsymbol{c_R} := \boldsymbol{M} \cdot \boldsymbol{b_R}$ and $\boldsymbol{s_R}' := \boldsymbol{s_R} + \boldsymbol{c_R}$
2. Refresh $\boldsymbol{s_L}$:
   - Sample two vectors $\boldsymbol{a_L}, \boldsymbol{b_L} \in Z_n^{*m} \backslash \{(0^m)\}$ such that $\boldsymbol{a_L} \cdot \boldsymbol{b_L} = 0$
   - Generate a random non-singular matrix $N \in Z_n^{*m \times m}$ such that $\boldsymbol{N} \cdot \boldsymbol{s_R}' = \boldsymbol{b_L}$
   - Set $\boldsymbol{c_L} := \boldsymbol{a_L} \cdot \boldsymbol{N}$ and $\boldsymbol{s_L}' = \boldsymbol{s_L} + \boldsymbol{c_L}$
3. Output $(\boldsymbol{s_L}', \boldsymbol{s_R}')$

The leakage-resilience of the refreshing protocol is stated in Theorem 2.

Theorem 2: Let $|Z_n| = \Omega(m)$ and $o(m) = 1$. $Refresh^m_{Z_n^*}(\boldsymbol{s_L}, \boldsymbol{s_R})$ is a $(l, 0.15 \cdot m \log |Z_n| - 1, negl(m))$-refreshing protocol for $Encode^m_{Z_n^*}(s)$, where l denotes the number of interactions between an adversary $\mathcal{A}$ and the refreshing experiment [35] and $negl(m)$ is some negligible function.

D. eCK Model

The eCK model [36] is a very strong security model for AKA protocols. It is defined by an eCK security experiment where an adversary $\mathcal{M}$ is given many corruption powers for various key agreement sessions and must solve a challenge on a test session. In the eCK security experiment, $\mathcal{M}$ controls a certain number of parties and plays with a challenger $\mathcal{C}$ through four phases as follows:

1. $\mathcal{M}$ makes any sequence of the following queries:
   - $Send(A, B, msg)$. $\mathcal{M}$ sends a message msg to party A on behalf of party B and is given the response according to the protocol specification.
   - $LongtermKeyReveal(A)$. $\mathcal{M}$ is given the long-term key of A.
   - $EphemeralKeyReveal(A, B, sid)$. $\mathcal{M}$ is given the ephemeral key of A and B in session sid.
   - $Reveal(sid)$. $\mathcal{M}$ is given a session key of a completed session sid.
2. $\mathcal{M}$ selects a completed session sid and makes the following query:
   - $Test(sid)$. $\mathcal{M}$ is given a challenge C which equals to $Reveal(sid)$ if $b = 1$ and is a random string if $b = 0$ for some randomly picked $b \in \{0, 1\}$.
3. $\mathcal{M}$ makes queries as in Phase 1.
4. $\mathcal{M}$ makes the following query and the experiment terminates:
   - $Guess(b')$. $\mathcal{M}$ submits the value of b'.

The selected test session sid is clean if all of the following conditions hold:

- $\mathcal{M}$ did not make $Reveal(sid)$ or $Reveal(sid^*)$ (if sid^* exists) where sid^* denotes the matching session to sid.
- sid^* exists and $\mathcal{M}$ did not make any set of the following queries:
  - $LongtermKeyReveal(A)$ and $EphemeralKeyReveal(A, B, sid)$
  - $LongtermKeyReveal(A)$ and $EphemeralKeyReveal(B, A, sid^*)$
  - $LongtermKeyReveal(B)$ and $EphemeralKeyReveal(A, B.sid)$
  - $LongtermKeyReveal(B)$ and $EphemeralKeyReveal(B, A, sid^*)$
- sid^* does not exist and $\mathcal{M}$ made any set of the following queries:
  - $LongtermKeyReveal(A)$ and $EphemeralKeyReveal(A, B, sid)$
  - $LongtermKeyReveal(B)$

$\mathcal{M}$ wins the eCK security experiment if $b' = b$ and the test session is clean.

Definition 1 (eCK security): An AKA protocol P is eCK-secure if no efficient adversary $\mathcal{M}$ wins the eCK security experiment with a nonnegligible advantage defined as

$$Adv_P^{eCK}(\mathcal{M}) = Pr[\mathcal{M}\ wins] - \frac{1}{2}.$$

The eCK model covers most passive and active attacks but not the side-channel attacks. It allows the attacker to reveal either long-term or ephemeral keys of a party, but not both. However, a side-channel attacker can leak both long-term and ephemeral keys of a party at the same time.

E. CAFL-eCK Model

The CAFL-eCK model [37] is extended from eCK model which captures side-channel attacks. It is defined by the following CAFL-eCK experiment where the adversary $\mathcal{M}$ is given all the power in eCK experiment as well as the power of side-channel attack:

1. $\mathcal{M}$ makes any sequence of the following queries:
   - $Send(A, B, msg, f)$. $\mathcal{M}$ sends a message msg to party A on behalf of party B and is given the response according to the protocol specification, along with the leakage $f(sk_A)$ where sk_A is the long-term secret key of A and f is a leakage function.
   - $Corrupt(A)$. This query is the same as the $LongtermKeyReveal(A)$ query in eCK secure experiment. $\mathcal{M}$ is given the long-term key of A.
   - $EphemeralKeyReveal(A, B, sid)$. This query is the same as the $EphemeralKeyReveal(A, B, sid)$ query in eCK secure experiment. $\mathcal{M}$ is given the ephemeral key of A and B in session sid.
   - $Reveal(sid)$. This query is the same as the $Reveal(sid)$ query in eCK secure experiment. $\mathcal{M}$ is given a session key of a completed session sid.
2. $\mathcal{M}$ selects a completed session sid and makes the following query:
   - $Test(sid)$. This query is the same as the $Test(sid)$ query in eCK secure experiment. $\mathcal{M}$ is given a challenge C which equals to $Reveal(sid)$ if $b = 1$ and is a random string if $b = 0$ for some randomly picked $b \in \{0, 1\}$.
3. $\mathcal{M}$ makes queries as in Phase 1.
4. $\mathcal{M}$ makes the following query and the experiment terminates:
   - $Guess(b')$. $\mathcal{M}$ submits the value of b'.

The selected test session is λ-CAFL-eCK-fresh if all the following conditions hold:

- $\mathcal{M}$ did not make $Reveal(sid)$ or $Reveal(sid^*)$ (if sid^* exists)
- sid^* exists and $\mathcal{M}$ did not make any set of the following queries:
  - $Corrupt(A)$ and $EphemeralKeyReveal(A, B, sid)$
  - $Corrupt(B)$ and $EphemeralKeyReveal(B, A, sid^*)$
- sid^* does not exist and $\mathcal{M}$ did not make any set of the following queries:
  - $Corrupt(A)$ and $EphemeralKeyReveal(A, B, sid)$
  - $Corrupt(B)$
- For each $Send(A, B, msg, f)$ query, $|f(sk_A)| \leq \lambda$

$\mathcal{M}$ wins the CAFL-eCK security experiment if $b' = b$ and the test session is λ-CAFL-eCK-fresh.

Definition 2 (λ-CAFL-eCK security): An AKA protocol P is λ-CAFL-eCK-secure if no efficient adversary $\mathcal{M}$ wins the

CAFL-eCK security experiment with a nonnegligible advantage defined as

$$Adv_P^{\lambda-CAFL-eCK}(\mathcal{M}) = Pr[\mathcal{M}\ wins] - \frac{1}{2}.$$

Let P be an AKA protocol constructed from an AKA protocol P' using a leakage-resilient encoding protocol $Encode_{Z_n^*}^m$ and a leakage-resilient refreshing protocol $Refresh_{Z_n^*}^m$. The following theorem shows the relation between the CAFL-eCK security of P and the eCK security of P' [37]. It will be used in the security proof of the proposed Protocol II.

Theorem 3: Protocol P is λ-CAFL-eCK-secure if the underlying protocol P' is eCK-secure, and $Refresh_{Z_n^*}^m$ is a (l, λ, ϵ)-refreshing protocol for $Encode_{Z_n^*}^m$. The advantage $\mathrm{Adv}_P^{\lambda-\mathrm{CAFL-eCK}}(\mathcal{M})$ of a PPT adversary $\mathcal{M}$ against P in the λ-CAFL-eCK security game is $\leq n_p(\mathrm{Adv}_{P'}^{\mathrm{eCK}}(\mathcal{M}) + \epsilon)$, where n_p denotes the number of parties in the eCK and CAFL-eCK security experiments.

IV. System Model

A. Network Framework

The blockchain-powered intelligent edge framework is shown in Fig 1. It has three layers:

- The end layer consists of IoT end devices which collect data from the environment (e.g., temperature sensors) or execute instructions (e.g., intelligent lamps).
- The edge layer consists of edge devices which mainly transmit data between the end layer and the cloud layer. It also processes some data on behalf of the cloud layer and maintains blockchain ledger to keep track of data processing.
- The cloud layer is composed of cloud servers which are used to store and process large volumes of data. It can trace data processed in the edge layer through the blockchain ledger.

There are three types of entities in the blockchain-powered intelligent system. The network model among them is abstracted in Fig 2.

- The channels between KGC (cloud centers) and users (edge/end devices) are secure channels during user registration and key extraction processes. The secure channel is only used once in the initialization process (i.e., key extraction). It is not used in the AKA processes. It can be realized via out-of-band manners (e.g., QR code) or a secure handshake. We will illustrate one method in the use case in Section VIII.
- The channels between users are public channels during AKA process. These channels are subject to attacks explained below in Section IV-B.

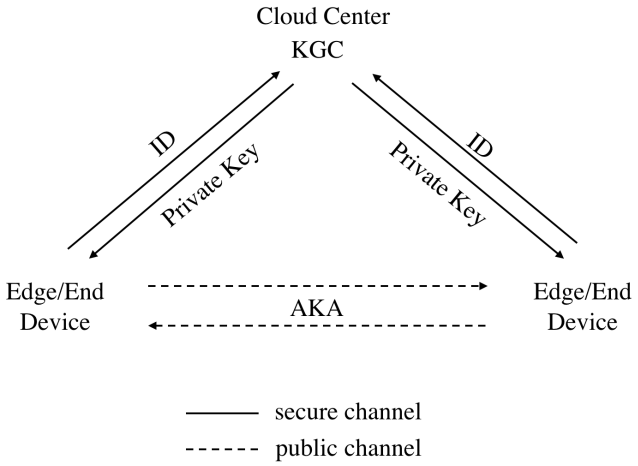


Fig. 2: Network framework

B. Threat Model

The attackers against the system are assumed to have the following powers:

- Controlling the message flow over public channels. This coresponses to $Send$ query in eCK and CAFL-eCK models;
- Revealing ephemeral keys. This corresponses to $EphemeralKeyReveal$ query in eCK and CAFL-eCK models;
- Revealing long-term keys. This coresponses to $Corrupt$ query in eCK and CAFL-eCK models;
- Side-channel attacks. This coresponses to $Send$ query in CAFL-eCK model.

The restriction is that the attackers are not allowed to reveal both ephemeral and long-term secrets of the same party, since in this case, they will trivially break any AKA protocol.

C. Design Objectives

Under the above threat model, the proposed key agreement protocols are expected to achieve the following objectives:

- eCK security which guarantees mutual authentication and agreement of secure session keys between two parties against attackers with above powers excluding side-channel attacks.
- CAFL-eCK security which guarantees mutual authentication and agreement of secure session keys between two parties against attackers with above powers including side-channel attacks.
- Affordable computing burden for IoT devices.

V. Leakage-Resilient ID-Based Key Agreement

This section presents two ID-based key agreement protocols. It first gives an overview of these protocols, and then formally describes the two protocols: eCK secure Protocol I and CAFL-eCK secure Protocol II.

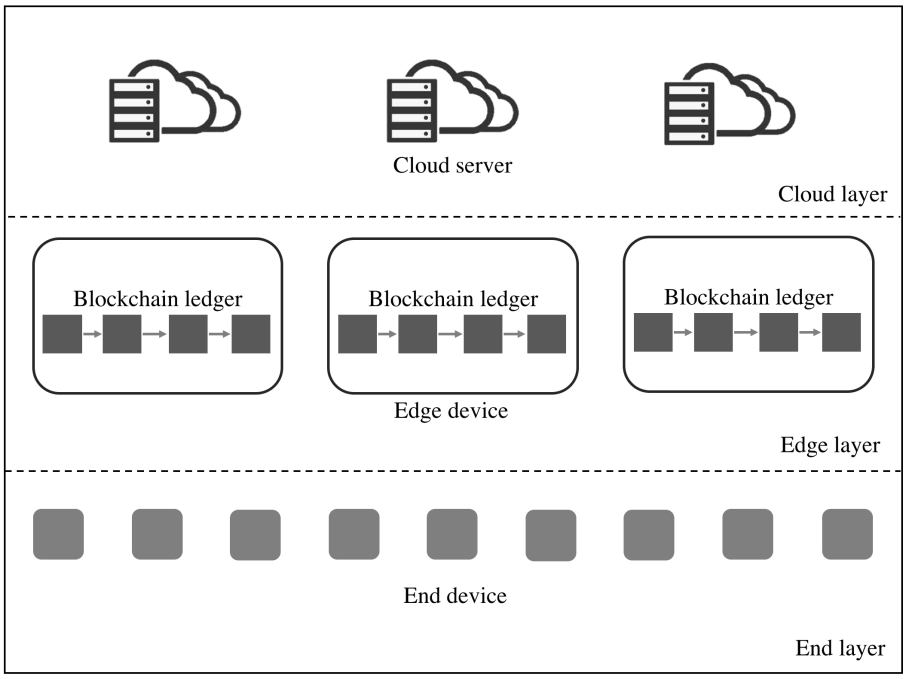


Fig. 1: System overview of the Blockchain-powered intelligent edge

A. Overview

1) Setup: KGC takes a security parameter $k \in Z^+$ as input and initializes system parameters and a master key:

1. Generate $\{p, F_p, G, P, H_1, H_2\}$ where p is a k-bit prime, F_p is a finite field of order p, G is an elliptic curve group over F_p, n is the order of G, P is a generator of G, $H_1 : \{0,1\}^* \times G \rightarrow Z_n$ and $H_2 : \{0,1\}^* \times \{0,1\}^* \times G \times G \times G \times G \rightarrow \{0,1\}^k$ are two cryptographic secure hash functions.
2. Select a random value x from Z_n as the master private key and compute the master public key $P_{pub} = [x]P$.
3. Output $\{p, F_p, G, P, H_1, H_2, P_{pub}\}$ as system parameters and secretly keep the master private key x.

2) Extract: KGC takes a user's identity ID as input and returns private key for the user:

1. Select a random r from Z_n and compute $R_{ID} = [r]P$.
2. Compute $h = H_1(ID||R_{ID})$.
3. Compute $s_{ID} = r + h \cdot x \mod n$.
4. Securely send the private key (R_{ID}, s_{ID}) to the user.

The user can verify the correctness of the private key through the following equation:

$$[s_{ID}]P \stackrel{?}{=} R_{ID} + [H_1(ID||R_{ID})]P_{pub}$$

3) AKA: Two communicating parties agree a fresh shared session key through the following phases:

1. Key exchange: Generate ephemeral secrets and public values, and exchange public values over open channels.
2. Shared secrets computation: Compute shared secrets using their own long-term and ephemeral secrets and the other party's public values.
3. Session key derivation: Derive the session key from the shared secrets using some key deriving function.

B. Protocol I: IDAKA

1) Setup: KGC initializes the system parameters $\{p, F_p, G, P, H_1, H_2, P_{pub}\}$ and the master private key x through the processes in Section V-A1.

2) Extract: Let A and B denote two communicating parties and their identities respectively. A and B extract their long-term private keys (R_A, s_A) and (R_B, s_B) from KGC through the processes in Section V-A2.

3) AKA: A and B agree the shared session key as follows:

1. Key exchange:
   - A selects a random a from Z_n, computes $T_A = [a]P$, and sends (A, R_A, T_A) to B.
   - B selects a random b from Z_n, computes $T_B = [b]P$, and sends (B, R_B, T_B) to A.
2. Shared secrets computation:
   - A computes the shared secrets as follows:

$$Z_{A1} = [s_A](R_B + [H_1(B||R_B)]P_{pub})$$
$$Z_{A2} = [a](R_B + [H_1(B||R_B)]P_{pub})$$
$$Z_{A3} = [s_A]T_B$$
$$Z_{A4} = [a]T_B$$

   - B computes the shared secrets as follows:

$$Z_{B1} = [s_B](R_A + [H_1(A||R_A)]P_{pub})$$
$$Z_{B2} = [s_B]T_A$$
$$Z_{B3} = [b](R_A + [H_1(A||R_A)]P_{pub}$$
$$Z_{B4} = [b]T_A$$

3. Session key derivation
   - A computes $K_A = H_2(A, B, Z_{A1}, Z_{A2}, Z_{A3}, Z_{A4})$
   - B computes $K_B = H_2(A, B, Z_{B1}, Z_{B2}, Z_{B3}, Z_{B4})$

C. Protocol II: LR-IDAKA

1) Setup: KGC takes a security parameter $k \in Z^+$ as input and initializes system parameters and two states of the master private key.

1. This step is identical with step 1 in Section V-A1. It generates parameters $\{p, F_p, G, P, H_1, H_2\}$.
2. Select two random vectors $\boldsymbol{x_L}, \boldsymbol{x_R} \in Z_n^{*m} \backslash \{(0^m)\} (m \geq 2)$ as the master private key states and compute the master public key as follows:
$$P_{pub} = [\boldsymbol{x_L} \cdot \boldsymbol{x_R}]P$$
3. Output $\{p, F_p, G, P, H_1, H_2, P_{pub}\}$ as system parameters and secretly keep $\boldsymbol{x_L}$ and $\boldsymbol{x_R}$ in two separate memories.

2) Extract: KGC takes a user's identity ID as input and returns the private key for the user:

1. Select a random r_{ID} from Z_n and compute $R_{ID} = [r_{ID}]P$.
2. Compute $h = H_1(ID||R_{ID})$.
3. Compute $\boldsymbol{s_{ID}^1} = h \cdot \boldsymbol{x_L}$ and $s_{ID} = r_{ID} + \boldsymbol{s_{ID}^1} \cdot \boldsymbol{x_R}$.
4. Invoke $Refresh_{Z_n^*}^m(\boldsymbol{x_L}, \boldsymbol{x_R})$ to update the memories for $\boldsymbol{x_L}$ and $\boldsymbol{x_R}$.
5. Securely send the private key (R_{ID}, s_{ID}) to the user.

Specifically, A and B receive (R_A, s_A) and (R_B, s_B) respectively. Then they store their private keys through the leakage-resilient encoding protocol:

- A invokes $Encode_{Z_n^*}^m(s_A)$ to output $\boldsymbol{s_{AL}}$ and $\boldsymbol{s_{AR}}$, destroys s_A, and securely stores $\boldsymbol{s_{AL}}$ and $\boldsymbol{s_{AR}}$ in two separate memories.
- B invokes $Encode_{Z_n^*}^m(s_B)$ to output $\boldsymbol{s_{BL}}$ and $\boldsymbol{s_{BR}}$, destroys s_B, and securely stores $\boldsymbol{s_{BL}}$ and $\boldsymbol{s_{BR}}$ in two separate memories.

3) AKA: A and B agree a shared session key through the following steps.

1. Key exchange:
   - A selects a random a from Z_n, computes $T_A = [a]P$, and sends (A, R_A, T_A) to B.
   - B selects a random b from Z_n, computes $T_B = [b]P$, and sends (B, R_B, T_B) to A.
2. Shared secrets computation:
   - A computes the shared secrets as follows:
$$Z_{A1}^1 = [\boldsymbol{s_{AL}}](R_B + [H_1(B||R_B)]P_{pub})$$
$$Z_{A1} = [\boldsymbol{s_{AR}}]Z_{A1}^1$$
$$Z_{A2} = [a](R_B + [H_1(B||R_B)]P_{pub})$$
$$Z_{A3}^1 = [\boldsymbol{s_{AL}}]T_B$$
$$Z_{A3} = [\boldsymbol{s_{AR}}]Z_{A3}^1$$
$$Z_{A4} = [a]T_B$$
     and then invokes $Refresh_{Z_n^*}^m(\boldsymbol{s_{AL}}, \boldsymbol{s_{AR}})$ to update the memories for $\boldsymbol{s_{AL}}$ and $\boldsymbol{s_{AR}}$.
   - B computes the shared secrets as follows:
$$Z_{B1}^1 = [\boldsymbol{s_{BL}}](R_A + [H_1(A||R_A)]P_{pub})$$
$$Z_{B1} = [\boldsymbol{s_{BR}}]Z_{B1}^1$$
$$Z_{B2}^1 = [\boldsymbol{s_{BL}}]T_A$$
$$Z_{B2} = [\boldsymbol{s_{BR}}]Z_{B2}^1$$
$$Z_{B3} = [b](R_A + [H_1(A||R_A)]P_{pub}$$
$$Z_{B4} = [b]T_A$$
     and then invokes $Refresh_{Z_n^*}^m(\boldsymbol{s_{BL}}, \boldsymbol{s_{BR}})$ to update the memories for $\boldsymbol{s_{BL}}$ and $\boldsymbol{s_{BR}}$.
3. Session key derivation
   - A computes $K_A = H_2(A, B, Z_{A1}, Z_{A2}, Z_{A3}, Z_{A4})$
   - B computes $K_B = H_2(A, B, Z_{B1}, Z_{B2}, Z_{B3}, Z_{B4})$

D. Correctness

1) Correctness of Protocol I: By the end of Protocol I, the session keys computed by A and B are identical since

$$\begin{aligned} Z_{A1} &= [s_A](R_B + [H_1(B||R_B)]P_{pub}) \\ &= [s_A \cdot s_B]P = [s_B \cdot s_A]P \\ &= [s_B](R_A + [H_1(A||R_A)]P_{pub}) \\ &= Z_{B1} \end{aligned}$$

$$\begin{aligned} Z_{A2} &= [a](R_B + [H_1(B||R_B)]P_{pub}) \\ &= [a \cdot s_B]P = [s_B \cdot a]P = [s_B]T_A \\ &= Z_{B2} \end{aligned}$$

$$\begin{aligned} Z_{A3} &= [s_A]T_B \\ &= [s_A \cdot b]P = [b \cdot s_A]P \\ &= [b](R_A + [H_1(A||R_A)]P_{pub}) \\ &= Z_{B3} \end{aligned}$$

$$\begin{aligned} Z_{A4} &= [a]T_B \\ &= [a \cdot b]P = [b \cdot a]P = [b]T_A \\ &= Z_{B4} \end{aligned}$$

Hence we have

$$\begin{aligned} K_A &= H_2(A, B, Z_{A1}, Z_{A2}, Z_{A3}, Z_{A4}) \\ &= H_2(A, B, Z_{B1}, Z_{B2}, Z_{B3}, Z_{B4}) \\ &= K_B \end{aligned}$$

2) Correctness of Protocol II: By the end of Protocol I, the session keys computed by A and B are identical since

$$\begin{aligned} Z_{A1} &= [\boldsymbol{s_{AR}}]Z_{A1}^1 \\ &= [\boldsymbol{s_{AR}}]([\boldsymbol{s_{AL}}](R_B + [H_1(B||R_B)]P_{pub})) \\ &= [\boldsymbol{s_{AR}} \cdot \boldsymbol{s_{AL}}](R_B + [H_1(B||R_B)]P_{pub}) \\ &= [s_A](R_B + [H_1(B||R_B)]P_{pub}) \\ &= [s_A][s_B]P = [s_B][s_A]P \\ &= [s_B](R_A + [H_1(A||R_A)]P_{pub}) \\ &= [\boldsymbol{s_{BR}} \cdot \boldsymbol{s_{BL}}](R_A + [H_1(A||R_A)]P_{pub}) \\ &= [\boldsymbol{s_{BR}}]([\boldsymbol{s_{BL}}](R_A + [H_1(A||R_A)]P_{pub})) \\ &= [\boldsymbol{s_{BR}}]Z_{B1}^1 \\ &= Z_{B1} \end{aligned}$$

$$\begin{aligned} Z_{A2} &= [a](R_B + [H_1(B||R_B)]P_{pub}) \\ &= [a \cdot s_B]P = [s_B \cdot a]P = [s_B]T_A \\ &= [\boldsymbol{s_{BR}} \cdot \boldsymbol{s_{BL}}]T_A = [\boldsymbol{s_{BR}}][\boldsymbol{s_{BL}}]T_A \\ &= [\boldsymbol{s_{BR}}]Z^1_{B2} \\ &= Z_{B2} \end{aligned}$$

$$\begin{aligned} Z_{A3} &= [\boldsymbol{s_{AR}}]Z^1_{A3} \\ &= [\boldsymbol{s_{AR}}]([\boldsymbol{s_{AL}}]T_B) = [\boldsymbol{s_{AR}} \cdot \boldsymbol{s_{AL}}]T_B \\ &= [s_A]T_B = [s_A \cdot b]P = [b \cdot s_A]P \\ &= [b](R_A + [H_1(A||R_A)]P_{pub}) \\ &= Z_{B3} \end{aligned}$$

$$\begin{aligned} Z_{A4} &= [a]T_B \\ &= [a \cdot b]P = [b \cdot a]P = [b]T_A \\ &= Z_{B4} \end{aligned}$$

Therefore, the session key

$$\begin{aligned} K_A &= H_2(A, B, Z_{A1}, Z_{A2}, Z_{A3}, Z_{A4}) \\ &= H_2(A, B, Z_{B1}, Z_{B2}, Z_{B3}, Z_{B4}) \\ &= K_B \end{aligned}$$

VI. Security Proof

This section proves the eCK security of Protocol I and CAFL-eCK security of Protocol II.

A. eCK security of Protocol I

The eCK security of Protocol I is declared in Theorem 4 which is proved via the eCK security experiment introduced in Section III-D.

Theorem 4: Protocol I is eCK secure if H_2 is modeled as a random oracle and the ECGDH problem is hard in G.

For any adversary $\mathcal{M}$ against Protocol I that involves at most n_p honest parties and activates at most n_s sessions, the advantage of $\mathcal{M}$ is:

$$Adv_I^{eCK}(\mathcal{M}) \leq \max(\frac{n_s^2}{2}, n_p n_s) \cdot Adv_G^{ECGDH}(\mathcal{S})$$

where $\mathcal{S}$ denotes a ECGDH solver which is constructed using $\mathcal{A}$ and against the ECGDH problem in G.

We give the proof sketch to Theorem 4 as follows. The complete security proof is provided in the appendix.

Proof Sketch: To prove Protocol I is eCK secure, we need to prove that no efficient adversary $\mathcal{M}$ wins the eCK security experiment with a nonnegligible advantage, that is,

$$Adv_I^{eCK}(\mathcal{M}) = Pr[\mathcal{M}\ wins] - \frac{1}{2}$$

is negligible for any adversary $\mathcal{M}$ against Protocol I.

We construct a ECGDH solver $\mathcal{S}$ against the ECGDH problem in G using $\mathcal{M}$ as a subroutine. The construction is introduced as follows. $\mathcal{S}$ takes an ECGDH challenge $V = [v]P, W = [w]P$ for some unknown $v, p \in Z_n$, plays the eCK experiment as a challenger with $\mathcal{M}$, and replaces some values returned by honest parties in a way such that 1) $\mathcal{M}$ cannot detect any modification made by $\mathcal{S}$, and 2) when $\mathcal{M}$ wins the experiment, $\mathcal{S}$ can output the result to $ECGDH(V, W)$. We can prove that

$$Adv_I^{eCK}(\mathcal{M}) \leq \max(\frac{n_s^2}{2}, n_p n_s) Adv_G^{ECGDH}(\mathcal{S})$$

Given that the ECGDH problem is hard in G, $Adv_G^{ECGDH}(\mathcal{S})$ is negligible. Therefore, $Adv_I^{eCK}(\mathcal{M})$ is negligible.

■

B. λ-CAFL-eCK security of Protocol II

The CAFL-eCK security of Protocol II is declared in Theorem 5 which is proved via the eCK security of Protocol I and Theorem 3.

Theorem 5: Protocol II is λ-CAFL-eCK secure if H_2 is modeled as a random oracle and the ECGDH problem is hard in G. For any adversary $\mathcal{M}$ against Protocol II, that involves at most n_p honest parties and activates at most n_s sessions, the advantage of $\mathcal{M}$ is:

$$Adv_{II}^{\lambda-CAFL-eCK}(\mathcal{M}) \leq n_p \cdot (Adv_I^{eCK}(\mathcal{M}) + \epsilon).$$

where ϵ is negligible.

Proof: Protocol II is constructed based on Protocol I and $Encode^m_{Z^*_n}$ and $Refresh^m_{Z^*_n}$. First, Protocol I is eCK secure, which has been proved in Theorem 4. Second, Theorem 1 and 2 show that $Refresh^m_{Z^*_n}$ is a (l, λ, ϵ)-refreshing protocol for $Encode^m_{Z^*_n}$ where $\lambda = 0.15 \cdot m \log n$. Finally, according to Theorem 3,

$$Adv_{II}^{\lambda-CAFL-eCK}(\mathcal{M}) \leq n_p \cdot (Adv_I^{eCK}(\mathcal{M}) + \epsilon)$$

is negligible since $Adv_I^{eCK}(\mathcal{M})$ and ϵ are negligible. Therefore, Protocol II is CAFL-eCK secure according to Definition 2.

■

VII. Performance

This section presents theoretical evaluation and experimental tests of the performance of the proposed protocols.

A. Evaluation

The performance is evaluated from two aspects: computing cost and secure storage cost. The computing cost is evaluated via the number of time-consuming operations, i.e., scalar multiplications. The secure storage cost is evaluated via the size of secret keys. The results are listed in Table III and IV where $m \geq 2$ for security purpose and l denotes the length of a private key.

From Table III we can find that:

- The computing costs on KGC in Protocol II will be higher than in Protocol I. Although Protocol I and II require equal numbers of scalar multiplications, Protocol II requires additional execution of the $Refresh$ protocol in the extract process.
- The computing costs on users A and B in Protocol II are higher than in Protocol I. Specifically, Protocol II requires $4m-2$ scalar multiplications more than Protocol I on both

TABLE III: Evaluation of computing cost

Protocol	Stage	Scalar mulitplication on KGC	Scalar mulitplication on A	Scalar mulitplication on B
Protocol I	Setup	1	0	0
	Extract	1	0	0
	AKA	0	6	6
Protocol II	Setup	1	0	0
	Extract	1	0	0
	AKA	0	$4m+4$ $(m \geq 2)$	$4m+4$ $(m \geq 2)$

TABLE IV: Evaluation of secure storage cost

Protocol	Secret key size on KGC	Secret key size on A	Secret key size on B
Protocol I	l	l	l
Protocol II	$2ml$ $(m \geq 2)$	$2ml$ $(m \geq 2)$	$2ml$ $(m \geq 2)$

A and B. Given that $m \geq 2$, the increased computing cost is at least 6 scalar multiplications on each user.

From Table IV, we can see that the secure storage costs on KGC and the two users in Protocol II are higher than in Protocol I. The increased cost is $(2m-1)l$, i.e., at least $3l$ on both KGC and the two parties.

Overall, Protocol II has lower performance than Protocol I. This is reasonable as Protocol II has stronger security and can resist side-channel attacks. The next subsection will study whether the costs of the proposed protocols are acceptable for some widely used devices in the blockchain-powered intelligent edge environments.

B. Experiments

Prototypes of the proposed protocols are realized with the parameters specified in Table VI. They are run on the experimental platform introduced in Table V.

TABLE VI: Prototype parameters

Parameters	Specification
Elliptic curve	P-192 and P-256 in NIST.FIPS.186-3
H_1, H_2, H_3	SHA-256 in NIST. FIPS.180-2
l	192 bits, 256 bits
m	2
Enc, Dec	Rabbit stream cipher in RFC 4503

In the experiments, the computing cost is tested via average computing time on each device. Each protocol is run 10 times to test the computing time. The average computing time is compared in Fig. 3.

From the comparison we can see that:

- For each of the three devices, the computing time in Protocol II is higher than in Protocol I.
- The computing time of Protocol II with P-192 and P-256 is acceptable for commonly used devices. The longest computing time is 1.56 seconds with P-256 on a Raspberry Pi.

The secure storage cost is tested via the files size of private keys on each device. The file size is compared in Fig. 4.

From the comparison we conclude that:

- For each of the three devices, the increased secure storage in Protocol II is about 3 times that of in Protocol I.
- The requirement for secure storage of Protocol II with P-192 and P-256 are acceptable for widely used devices. The largest secure storage for A or B is 467 bytes with P-256. This is acceptable for Raspberry Pi.

In summary, although Protocol II has higher computing and secure storage costs than Protocol I, the costs are acceptable for widely used devices in the blockchain-powered intelligent edge.

C. Comparison

Table I in Section II-A6 has shown comparisons of the proposed protocols with some existing ID-based AKA protocols in terms of efficiency, leakage-resiliency, and provable security. It shows that Protocol I has the strongest provable security among all non-leakage-resilient protocols, and Protocol II has the strongest provable security among all leakage-resilient ones.

Here we use Table VII to compare the peformance of the proposed protocols with some representive ID-based AKA protocols (pairing-free and/or leakage-resilient ones) which can be used in the same scenario. In the table, $\mathcal{E}$ denotes group exponentiation (or scalar multiplication in elliptic curve group), and $\mathcal{P}$ denotes bilinear pairing. The computing cost on a user is evaluated by the number of time-consuming operations during key agreement process.

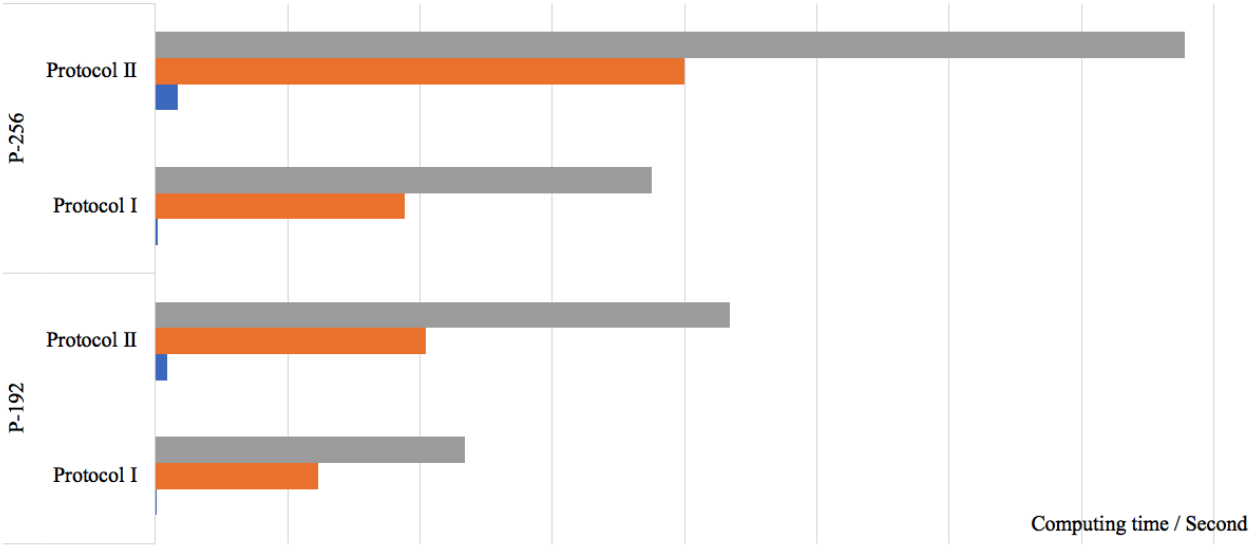
TABLE VII: Comparison for performance

Protocol	Computing cost on a user	Security
Protocol I	$6\mathcal{E}$	eCK
Protocol II	$(4m+4)\mathcal{E}$ $(m \geq 2)$	CAFL-eCK
[14]	$4\mathcal{E}$	mBR
[15]	$6\mathcal{E}$	eCK
[16]	$4\mathcal{E}$	Modified eCK
[25]	$4\mathcal{E}$	CK
[26]	$3\mathcal{E}$	CK
[27]	$7\mathcal{E}$	eCK
[28]	$5\mathcal{E}$	$\times$
[29]	$2\mathcal{E}$	mBR
[11]	$8\mathcal{E}+2\mathcal{P}$	BAFL-eCK
[12]	$4\mathcal{E}+3\mathcal{P}$	CL-eCK

According to Table VII, among all non-leakage-resilient and pairing-free protocols, the computing cost of Protocol I is equal to or lower than that of the other protocols ([15], [27])

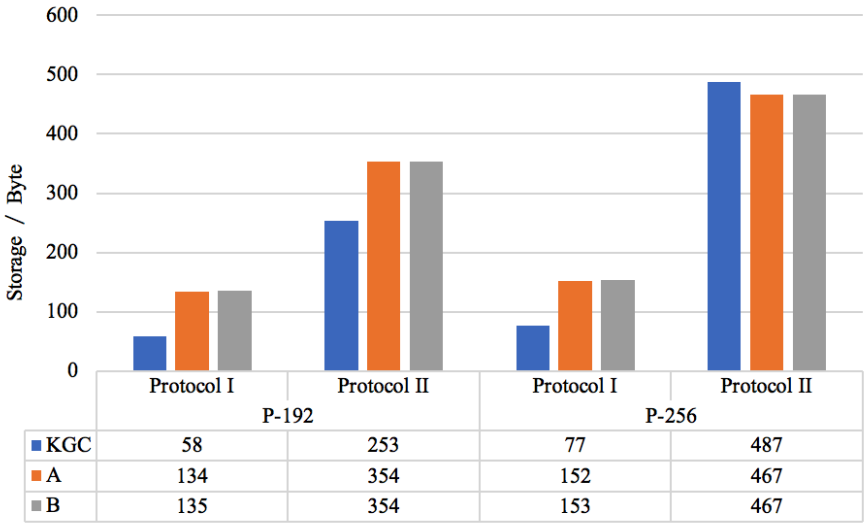
TABLE V: Experimental platform

Participant	Device	CPU	Memory	Storage
Cloud server (KGC)	Desktop	3.2 GHz Intel Core i5	8 GB	921 GB
Edge device (A or B)	Laptop	2.3 GHz Intel Core i5	8 GB	256 GB
End device (A or B)	Raspberry Pi	1.5 GHz Cortex A-72	2 GB	16 GB



	P-192		P-256	
	Protocol I	Protocol II	Protocol I	Protocol II
End device	0.4678	0.8682	0.7510	1.5556
Edge device	0.2467	0.4084	0.3777	0.7991
KGC	0.0027	0.0184	0.0045	0.0342

Fig. 3: Experiment result: average computing time on each device



	P-192		P-256	
	Protocol I	Protocol II	Protocol I	Protocol II
KGC	58	253	77	487
A	134	354	152	467
B	135	354	153	467

Fig. 4: Experiment result: secure storage on each device

with same level of security (eCK). Protocol II is the only pairing-free protocol with leakage-resiliency; and its security is the strongest among all leakage-resilient protocols.

VIII. Use Case

This section demonstrates the application of the proposed ID-based key agreement protocols in the blockchain-powered

smart home scenario.

A. System Overview

The blockchain-powered smart home is shown in Fig. 5. It has three layers:

- The end layer is composed of smart end devices such as smart lamps, robot vacuums, cameras, etc. These devices have limited computing and networking capabilities.
- The edge layer is composed of intelligent edge devices such as laptops, tablets, smart phones, etc. These devices collect and process data from the end layer and communicate with the cloud layer. Besides, a blockchain ledger is maintained to keep track of data processing. The edge devices are much powerful than end devices in terms of computing and networking capabilities.
- The cloud layer is composed of powerful cloud servers which are used to store and process large volumes of data. It can trace data processed in the edge layer through the blockchain ledger.

B. Implementation of ID-based Key Agreement

This scenario assumes that the cloud center and edge devices are threaten by side-channel attacks, and the end devices in the smart home are safe from side-channel attacks. To achieve high efficiency and strong security, the ID-based key agreement mechanism uses a hybrid implementation of Protocol I and II.

1) Implementation of Setup: The leakage-resilient KGC of Protocol II is deployed on the cloud server. It extracts private keys for all edge and end devices in the smart home. The two vectors $\boldsymbol{x_L}$ and $\boldsymbol{x_R}$ of master private key are stored securely and separately.

2) Implementation of Extract: The Extract process is illustrated in Fig. 6. A device (e.g. a smart lamp) registers to the cloud center and extracts its private key via the following steps:

- The smart lamp generates a random $a \leftarrow Z_n$, computes $A = [a]P$ and sends A and its identity $lamp001$ to the cloud center (KGC).
- The cloud center generates a random $c \leftarrow Z_n$, and computes $C = [c]P$ and $k_{AC} = H_3([c]A)$ where H_3 is a cryptography hash function. Then it generates the private key $(R_{lamp001}, s_{lamp001})$ for the smart lamp, encrypts the private key using k_{AC} and sends the cyphertext $E_{lamp001}$, C and its identity ID_{KGC} to the smart lamp.
- The smart lamp first computes $k_{AC} = H_3([a]C)$. Then it decrypts $E_{lamp001}$ and acquires its private key.

In the above process, a secure link is established between the end device and cloud center to transmit the private key using Elliptic Curve Diffie-Hellman (ECDH) key exchange. Note that this is only run once in the extract procedure and is not executed in every run of the AKA procedure. Besides, due to space limitations we did not involve authentication in the above process. In real-world applications of smart home, authentication is easy to establish with the assistance of human users.

3) Implementation of AKA: The hybrid implementation of AKA processes of Protocol I and II is as follows:

- Smart edge devices: AKA of Protocol II is deployed on the edge devices. After extracting the private key from KGC, two vectors for the private key are generated and stored securely and separately in the edge device.
- Smart end devices: AKA of Protocol I is deployed on the end devices. After extract, the private key is stored securely in the end device.

4) Testing: A prototype is realized and tested on the platform specified in Table V. The elliptic curve used here is P-256, the three hash functions are SHA-256, the parameter $m = 2$, the encryption (decryption) algorithm is Rabit stream cipher in RFC 4503. After setup and extract, the AKA procedures are executed 10 times between the end and edge devices. The results are shown in Fig. 7.

We can draw the following conclusions from Fig. 7: 1) The storage cost on the end device is acceptable. The size of compiled protocol file is 4045 bytes, and the size of secret keys file is 231 bytes. Both are much smaller than the storage of commonly used smart end devices (up to 32 GB or 64 GB). 2) The computing costs are also acceptable for the limited end device. The computing time of devices registration and private key extract is only 0.13 seconds on the end device, and that of AKA is only 0.75 seconds.

IX. Conclusion

We have investigated secure and efficient key agreement for the blockchain-powered intelligent edge. Two ID-based AKA protocols with eCK security and CAFL-eCK security respectively have been presented. Compared with existing key agreement protocols, the proposed protocols are more convenient and efficient for the blockchain-powered intelligent edge environment. Besides, Protocol II can resist side-channel attacks to both KGC and the communicating parties. A use case which makes use of a hybrid implementation of the proposed two protocols in blockchain-powered smart home has been demonstrated. For future work we will consider how to further improve the storage and computation efficiency of the leakage resilient ID-based AKA protocols.

Appendix
Proof of Protocol I

Proof: Firstly, we represent the computation of session key as follows:

$$\begin{aligned}
K &= H_2(A, B, Z_{A1}, Z_{A2}, Z_{A3}, Z_{A4}) \\
&= H_2(A, B, Z_{B1}, Z_{B2}, Z_{B3}, Z_{B4}) \\
&= H_2(A, B, CDH(PK_A, PK_B), CDH(T_A, PK_B), \\
&CDH(PK_A, T_B), CDH(T_A, T_B)) \\
&= H_2(\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6) \\
&= H_2(\sigma)
\end{aligned}$$

where $PK_A = [s_A]P$ and $PK_B = [s_B]P$ are the long-term public keys of A and B, and $T_A = [a]P$ and $T_B = [b]P$ are the ephemeral public keys of A and B.

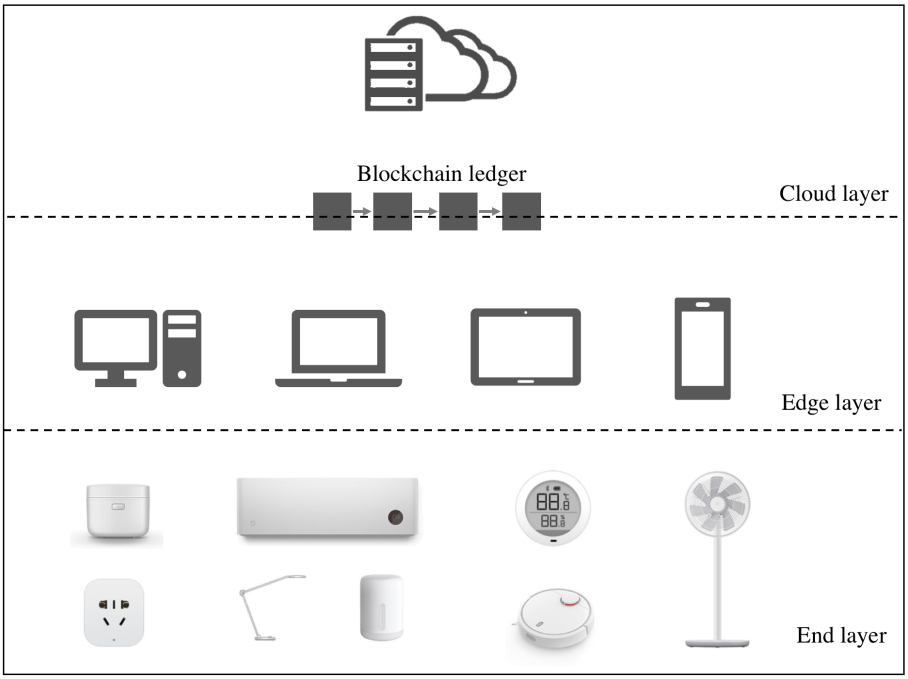


Fig. 5: System overview of the blockchain-powered smart home

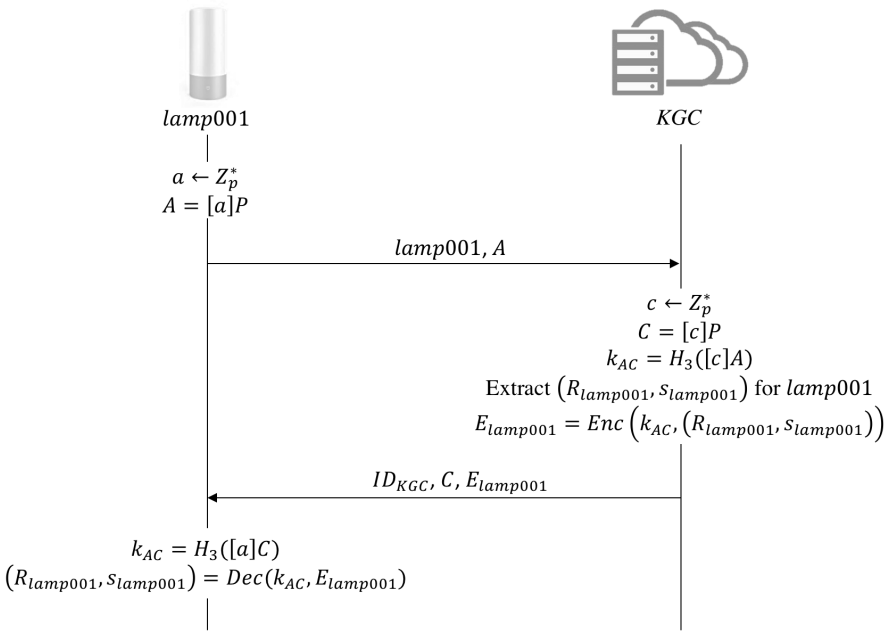


Fig. 6: Device registration and private key extract. Note that this process is only executed once and is not executed in AKA procedures.

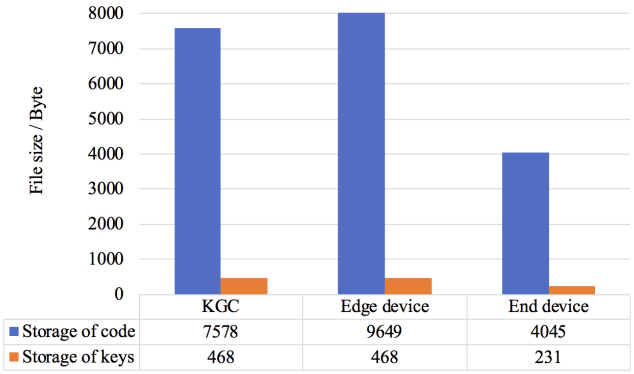
To win the eCK security experiment, $\mathcal{M}$ should successfully distinguish the session key K of the test session from a random string of the same length. There are two ways for the event $\mathcal{M}$ $wins$ happens:

- $\mathcal{M}$ correctly guesses the result. The probability is 1/2.
- $\mathcal{M}$ queries H_2 on the same 6-tuple σ, denoted by event E.

Therefore,

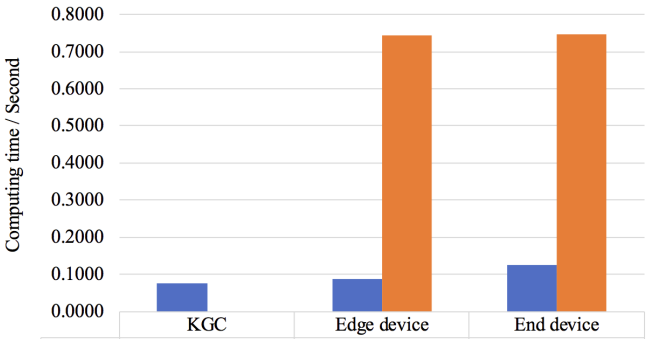
$$Pr[\mathcal{M}\ wins] = \frac{1}{2} + Pr[E]$$

If event E happens, we can construct an $ECGDH$ solver $\mathcal{S}$ against the ECGDH problem in G and using $\mathcal{M}$ as a subroutine. Specifically, $\mathcal{S}$ takes an ECGDH challenge $V = [v]P, W = [w]P$ for some unknown $v, p \in Z_n$, plays the eCK experiment as a challenger with $\mathcal{M}$, and replaces some values returned by honest parties in a way such that



	KGC	Edge device	End device
Storage of code	7578	9649	4045
Storage of keys	468	468	231

(a) Storage costs.



	KGC	Edge device	End device
Extract	0.0752	0.0882	0.1256
AKA		0.7423	0.7474

(b) Computing costs.

Fig. 7: Costs in the use case

- $\mathcal{M}$ cannot detect any modification made by $\mathcal{S}$, and
- when $\mathcal{M}$ wins the experiment, $\mathcal{S}$ can output the result to $ECGDH(V, W)$.

$\mathcal{S}$ is constructed as follows in two different cases:

- Case 1: the test session has a matching session
- Case 2: the test session does not have a matching session

In case 1, $\mathcal{S}$ randomly chooses a pair of matching sessions sid and sid^* involving two parties A and B, and guesses that one of them is the test session. $\mathcal{S}$ make the following modifications in the experiment:

- If $\mathcal{M}$ queried both $LongtermKeyReveal(A)$ and $LongtermKeyReveal(B)$, $\mathcal{S}$ can replace T_A with V and T_B with W in the experiment. This wont be found by $\mathcal{M}$ as $\mathcal{M}$ cannot query $EphemeralKeyReveal(A, B, sid)$ and $EphemeralKeyReveal(B, A, sid^*)$ to make sure the test session is clean. When $\mathcal{M}$ wins the experiment, $\mathcal{S}$ can output σ_6 as the result to $ECGDH(V, W)$.
- If $\mathcal{M}$ did not query either $LongtermKeyReveal(A)$ or $LongtermKeyReveal(B)$, $\mathcal{S}$ can replaces PK_A with V and PK_B with W in the experiment. When $\mathcal{M}$ wins the experiment, $\mathcal{S}$ can output σ_3 as the result to $ECGDH(V, W)$.
- If $\mathcal{M}$ queried $LongtermKeyReveal(A)$ and did not query $LongtermKeyReveal(B)$, $\mathcal{S}$ can replaces T_A with V and PK_B with W in the experiment. When $\mathcal{M}$ wins the experiment, $\mathcal{S}$ can output σ_5 as the result to $ECGDH(V, W)$.
- If $\mathcal{M}$ queried $LongtermKeyReveal(B)$ and did not query $LongtermKeyReveal(A)$, $\mathcal{S}$ can replaces T_B with V and PK_A with W in the experiment. When $\mathcal{M}$ wins the experiment, $\mathcal{S}$ can output σ_4 as the result to $ECGDH(V, W)$.

Denote the event that $\mathcal{S}$ successfully guessed the test session in case 1 by E_1, we have

$$Adv_G^{ECGDH}(\mathcal{S}) \geq \frac{2}{n_s^2} \cdot Pr[E_1]$$

thus

$$Pr[E_1] \leq \frac{n_s^2}{2} \cdot Adv_G^{ECGDH}(\mathcal{S})$$

In case 2, $\mathcal{S}$ randomly chooses a party B and a session sid where B is the responder, and guesses that sid is the test session. $\mathcal{S}$ make the following modifications in the experiment:

- If $\mathcal{M}$ queried $LongtermKeyReveal(A)$ for the initiator of sid, $\mathcal{S}$ has two choices to modify the experiment and output the result to $ECGDH(V, W)$:
  - replace T_B with V and PK_A with W, and output σ_4 when $\mathcal{M}$ wins
  - replace T_A with V and T_B with W, and output σ_6 when $\mathcal{M}$ wins
- If $\mathcal{M}$ did not queried $LongtermKeyReveal(A)$, $\mathcal{S}$ has two choices to modify the experiment and output the result to $ECGDH(V, W)$:
  - replace PK_A with V and PK_B with W, and output σ_3 when $\mathcal{M}$ wins
  - replace T_A with V and PK_B with W, and output σ_5 when $\mathcal{M}$ wins

Denote the event that $\mathcal{S}$ successfully guessed the party B and the test session in case 2 by E_2, we have

$$Adv_G^{ECGDH}(\mathcal{S}) \geq \frac{1}{n_p n_s} \cdot Pr[E_2]$$

thus

$$Pr[E_2] \leq n_p n_s \cdot Adv_G^{ECGDH}(\mathcal{S})$$

In summary,

$$\begin{aligned} Pr[E] &= \max(Pr[E_1], Pr[E_2]) \\ &\leq \max(\frac{n_s^2}{2}, n_p n_s) Adv_G^{ECGDH}(\mathcal{S}) \end{aligned}$$

According to Definition 1,

$$\begin{aligned} Adv_I^{eCK}(\mathcal{M}) &= Pr[\mathcal{M}\ win] - \frac{1}{2} \\ &= Pr[E] \\ &\leq \max(\frac{n_s^2}{2}, n_p n_s) Adv_G^{ECGDH}(\mathcal{S}) \end{aligned}$$

is negligible since ECGDH is hard in G. Therefore, Protocol I is eCK secure.

■

REFERENCES

[1] H. Liu, Y. Zhang, T. Yang, *Blockchain-Enabled Security in Electric Vehicles Cloud and Edge Computing*. IEEE Network, 32(3), pp.78-83. 2018.

[2] X. Xu, X. Zhang, H. Gao, Y. Xue, L. Qi, W. Dou, *BeCome: Blockchain-Enabled Computation Offloading for IoT in Mobile Edge Computing*. IEEE Transactions on Industrial Informatics, 16(6), pp.418741-95. 2020.

[3] K. Gai, Y. Wu, L. Zhu, L. Xu, Y. Zhang, *Permissioned Blockchain and Edge Computing Empowered Privacy-Preserving Smart Grid Networks*. IEEE Internet of Things Journal, 6(5), pp.7992-8004. 2019.

[4] J. Pan, J. Wang, A. Hester, I. Alqerm, Y. Liu and Y. Zhao, *EdgeChain: An Edge-IoT Framework and Prototype Based on Blockchain and Smart Contracts*. IEEE Internet of Things Journal, 6(3), pp.4719-4732. 2019.

[5] S. Tuli, R. Mahmud, S. Tuli, R. Buyya, *Fogbus: A Blockchain-Based Lightweight Framework for Edge and Fog Computing*. Journal of Systems and Software, 154, pp.22-36. 2019.

[6] S. Guo, X. Hu, S. Guo, X. Qiu, F. Qi, *Blockchain Meets Edge Computing: A Distributed and Trusted Authentication System*. IEEE Transactions on Industrial Informatics, 16(3), pp.1972-83. 2019.

[7] E. Rescorla, *The Transport Layer Security (TLS) Protocol Version 1.3*. IETF, RFC 8446, 2018.

[8] L. Lei, W. Zhang, Y. Wang, X. Wang X, *A Pairing-Free Identity-Based Handover AKE Protocol with Anonymity in the Heterogeneous Wireless Networks*. International Journal of Communication Systems, 32(12), pp.e4000. 2019.

[9] Q. Tan, Z. Zeng, K. Bu, K. Ren, *PhantomCache: Obfuscating Cache Conflicts with Localized Randomization*. Network and Distributed Systems Security (NDSS) Symposium 2021, 21-25 February 2021.

[10] J. Berger, A. Klein, B. Pinkas *Flaw Label: Exploiting IPv6 Flow Label*. 2020 IEEE Symposium on Security and Privacy (SP), 18-21 May 2020.

[11] O. Ruan, Y. Zhang, M. Zhang, J. Zhou, L. Harn, *After-The-Fact Leakage-Resilient Identity-Based Authenticated Key Exchange*. IEEE Systems Journal, 12(2). 2017.

[12] J. D. Wu, Y. M. Tseng, S. S. Huang, *An Identity-Based Authenticated Key Exchange Protocol Resilient to Continuous Key Leakage*. IEEE Systems Journal, 13(4), pp.3968-79. 2019.

[13] A. Shamir, *Identity-Based Cryptosystems and Signature Schemes*. Advances in Cryptology - Crypto'84, pp.47–53. Springer, Heidelberg, LNCS 196, 1984.

[14] X. Cao, W. Kou, X. Du, *A Pairing-Free Identity-Based Authenticated Key Agreement Protocol with Minimal Message Exchanges*. Information Sciences, 180(15), pp.2895-903, 2010.

[15] H. Sun, Q. Wen, H. Zhang, Z. Jin, *A Strongly Secure Identity-Based Authenticated Key Agreement Protocol without Pairings Under the GDH Assumption*. Security and Communication Networks, 8(17), pp.3167-3179. 2015.

[16] S. Bala, G. Sharma, A. K. Verma, *PF-ID-2PAKA: Pairing Free Identity-Based Two-Party Authenticated Key Agreement Protocol for Wireless Sensor Networks*. Wireless Personal Communications, 87(3), pp.995-1012. 2016.

[17] J. Zhang, X. Huang, W. Wang, Y. Yue, *Unbalancing Pairing-Free Identity-Based Authenticated Key Exchange Protocols for Disaster Scenarios*. IEEE Internet of Things Journal, 6(1), pp.878-90. 2018.

[18] E. Okamoto, *Proposal for Identity-Based Key Distribution Systems*. Electronics Letters 22.24, pp.1283-1284, 1986.

[19] C. G. Gunther, *An Identity-Based Key Exchange Protocol*. Advances in Cryptology - Eurocrypt'89, pp. 29-37. Springer-Verlag, 1989.

[20] C. Boyd, A. Mathuria, D. Stebila, *Protocols for Authentication and Key Establishment*. Heidelberg: Springer, 2003.

[21] D. Boneh, M. Franklin, *Identity-Based Encryption from the Weil Pairing*. Annual international cryptology conference 2001 Aug 19, pp. 213-229. Springer, Berlin, Heidelberg, 2001.

[22] N. P. Smart, *Identity-Based Authenticated Key Agreement Protocol Based on Weil Pairing*. Electronics letters, 38(13), pp.630-632, 2002.

[23] L. Chen, C. Kudla, *Identity Based Authenticated Key Agreement Protocols from Pairings*. 16th IEEE Computer Security Foundations Workshop, 2003, pp. 219-233. IEEE. 2003.

[24] N. McCullagh, P. S. Barreto, *A New Two-Party Identity-Based Authenticated Key Agreement*. Cryptographers' Track at the RSA Conference 2005 Feb 14, pp. 262-274. Springer, Berlin, Heidelberg. 2005.

[25] D. Fiore, R. Gennaro, *Making the Diffie-Hellman Protocol Identity-Based*. Cryptographers'Track at the RSA Conference 2010 Mar 1, pp.165-178. Springer, Berlin, Heidelberg. 2010.

[26] M. Xie, L. Wang, *One-Round Identity-Based Key Exchange with Perfect Forward Security*. Information Processing Letters, 112(14-15). pp.587-91. 2012.

[27] L. Ni, G. Chen, J. Li, Y. Hao, *Strongly Secure Identity-Based Authenticated Key Agreement Protocols without Bilinear Pairings*. Information Sciences, 367, pp.176-193. 2016.

[28] S. H. Islam, G. P.Biswas GP, *A Pairing-Free Identity-Based Two-Party Authenticated Key Agreement Protocol for Secure and Efficient Communication*. Journal of King Saud University-Computer and Information Sciences, 29(1), pp.63-73. 2017.

[29] M. Kumar, P. C. Saxena, *PF-AID-2KAP: Pairing-Free Authenticated Identity-Based Two-Party Key Agreement Protocol for Resource-Constrained Devices*. In International Conference on Futuristic Trends in Network and Communication Technologies 2018 Feb 9, pp.425-440. Springer, Singapore. 2018.

[30] K. Mahmood, J. Arshad, S. A. Chaudhry, S. Kumari, *An Enhanced Anonymous Identity-Based Key Agreement Protocol for Smart Grid Advanced Metering Infrastructure*. International Journal of Communication Systems, 32(16), pp.e4137. 2019.

[31] S. Patonico, A. Braeken, K. Steenhaut, *Identity-Based and Anonymous Key Agreement Protocol for Fog Computing Resistant in the Canetti–Krawczyk Security Model*. Wireless Networks, 11, pp.1-3. 2019.

[32] A. Hassan, A. A. Omala, M. Ali, C. Jin, F. Li, *Identity-Based User Authenticated Key Agreement Protocol for Multi-Server Environment with Anonymity*. Mobile Networks and Applications, 24(3), pp.890-902. 2019.

[33] X. Jia, D. He, N. Kumar, K. K. Choo, *A Provably Secure and Efficient Identity-Based Anonymous Authentication Scheme for Mobile Edge Computing*. IEEE Systems Journal, 14(1), pp.560-571. 2019.

[34] D. Hankerson, A. J. Menezes, S. Vanstone, *Guide to Elliptic Curve Cryptography*. Springer Science and Business Media, 2006.

[35] S. Dziembowski, S. Faust, *Leakage-Resilient Cryptography from the Inner-Product Extractor*. International Conference on the Theory and Application of Cryptology and Information Security 2011 Dec 4, pp.702-721. Springer, Berlin, Heidelberg. 2011.

[36] B. LaMacchia, K. Lauter, A. Mityagin, *Stronger Security of Authenticated Key Exchange*. In International conference on provable security, pp.1-16. Springer, Berlin, Heidelberg. 2007.

[37] J. Alawatugoda, D. Stebila, C. Boyd, *Continuous After-the-fact Leakage-Resilient eCK-secure Key Exchange*. In IMA international conference on cryptography and coding, pp.277-294. Springer, Cham. 2015.

[38] J. Alawatugoda, D. Stebila, and C. Boyd, *Modelling After-the-fact Leakage for Key Exchange*. In ASIACCS'14, 207–216, ACM, 2014.

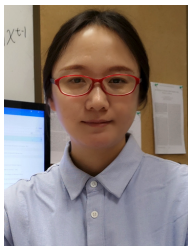
[39] M. Bellare, P. Rogaway, *Entity Authentication and Key Distribution*. In D. R. Stinson, editor, Advances in Cryptology - Crypto'93, pp. 232-249. Springer-Verlag, 1993. Lecture Notes in Computer Science Volume 773.

[40] M. Bellare, D. Pointcheval,P. Rogaway, *Authenticated Key Exchange Secure Against Dictionary Attacks*. In Advances in Cryptology - Crypto'00 (Lecture Notes in Computer Science), vol. 1807. Heidelberg,Germany: Springer, 2000, pp. 139–155.

[41] R. Canetti, H. Krawczyk, *Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels*. In: Pfitzmann B. (eds) Advances in Cryptology - EUROCRYPT 2001. Lecture Notes in Computer Science, vol 2045. Springer, Berlin, Heidelberg.

[42] L. Ni, G. Chen, J. Li, *Escrowable Identity-based Authenticated Key Agreement Protocol with Strong Security*. Computers & Mathematics with Applications, 65(9), pp. 1339-1349, Advanced Information Security, 2013

[43] D. Moriyama, T. Okamoto, *Leakage Resilient eCK-secure Key Exchange Protocol without Random Oracles*. ACM Symposium on Information, ACM, 2011:441.



Jie Zhang received the Ph.D. degree from Universtiy of Liverpool in 2018. Her research interests include cryptography and network security.

Futai Zhang received his bachelors and masters degrees in mathematics from Shaanxi Normal University, China, and the Ph. D degrees from Xidian University, China. His main research interests include cryptography and cyberspace security.